CSI5380: Systems and Architectures for Electronic Commerce

Project Part 1: "DreamTeam" CD Store – Design Documentation

Daniel Antwi

Khaled El Ghayesh

Elena Ciobanu

Cristian Gadea

October 2011

Contents

Figures

1. Introduction

The DreamTeam CD Store is an electronic commerce application that allows users to browse and order CDs. The CD Store was developed using technologies that include JSP (JavaServer Pages), Servlets, Web Services, and relational database management systems. Users are able to browse the list of available CDs in the music category of their choice once they have registered and logged in. The list of CDs and information about them is obtained from a relational database. Users can add CDs to a Shopping Cart, update the quantity of each CD in their cart, remove CDs from their cart, and view a summary of their order before they enter their credit card information and submit the order. The web application is secured using industry standard SSL (Secure Socket Layer) technology.

Details on building and deploying the project are provided in the included README.txt file.

2. Assumptions

The following is a list of some of the assumptions that were made when designing and developing this project:

- Little emphasis is placed on the visual aspect of the user interface (no images are used)
- Little validation is done on the client side to ensure users enter the proper values into forms (no JavaScript is used)
- Users cannot modify their account once it has been created
- A successful user order is shipped to the address with which the account was created

3. Project Files

This section contains a summary of all key files that make up the assignment source code. Additional project files used for building/deployment are described in the README.txt file.

3.1. Utilities

3.1.1. In **DBAgent/src** as part of **com.cdstore.dbagent**

DBAgent.java

- Uses JDBC to connect to the MySQL database to provide persistence-related actions such as checking the user's login information and retrieving the list of CDs

3.1.2. In **DBAgent/src**

queries.properties

- Configuration file containing the SQL queries to be executed on the database

3.1.3. In **Utilities/src** as part of **com.cdstore.entities**

Account.java

- Entity that defines Account objects as per database schema

Address.java

- Entity that defines Address objects as per database schema

Category.java

- Entity that defines Category objects as per database schema

CD.java

- Entity that defines CD objects as per database schema

Order.java

- Entity that defines Order objects as per database schema

OrderDetails.java

- Entity that defines OrderDetails objects as per database schema

3.1.4. In **Utilities/src** as part of **com.cdstore.shoppingcart**

ShoppingCart.java

- Code for the creation of shopping cart objects and retrieval/modification of items within them, reused from [1]

ShoppingCartItem.java

- ShoppingCartItem entity code for defining items that can be added to a shopping cart, reused from [1]

3.2. CatalogService

3.2.1. In **CatalogService/src** as part of **com.cdstore.catalogservice**

ProductCatalog.java

- The Web Service interface class of the product catalog that is implemented by the actual implementation class as well as used by the web client for retrieving CD information

ProductCatalogImpl.java

- The Web Service implementation of the product catalog that uses the DBAgent to fulfill services requested via SOAP (Simple Object Access Protocol)

3.2.2. In CatalogService/WebContent/WEB-INF

sun-jaxws.xml

- File used to define Web Service endpoints to the package containing the catalog Web Services code

web.xml

- deployment description file for the catalog Web Service

3.3. OrderService

3.3.1. In OrderService/src as part of com.cdstore.orderprocess

OrderProcess.java

- The Web Service interface class of the order process that is implemented by the actual implementation class as well as used by the web client for managing accounts and orders

OrderProcessImpl.java

- The Web Service implementation of the order process that uses the DBAgent to fulfill services requested via SOAP

3.3.2. In OrderService/WebContent/WEB-INF

sun-jaxws.xml

- File used to define Web Service endpoints to the package containing the order process Web Services code

web.xml

- deployment description file for the catalog Web Service

3.4. CDStore

3.4.1. In CDStore/src as part of com.cdstore.beans

CDStoreBean.java

- Backing bean used by the session controller to communicate with the product catalog Web Services

OrderBean.java

- Backing bean used by the session controller to communicate with the order process Web Services

3.4.2. In **CDStore/src** as part of **com.cdstore.catalogservice**

ProductCatalog.java

- The Web Service interface class of the product catalog that is required for using the catalog web service

ProductCatalogServiceClient.java

- Class that provides a getServiceInterface method for the session controller to call when accessing the product catalog Web Services

3.4.3. In **CDStore/src** as part of **com.cdstore.controller**

SessionController.java

- The session controller handles the user inputs and uses the Web Services to access the persisted data and display the appropriate HTML content to the user

3.4.4. In **CDStore/src** as part of **com.cdstore.filter**

SessionController.java

- The session filter is used to ensure only logged in users (who have a valid session) are able to access certain JSP pages

3.4.5. In **CDStore/src** as part of **com.cdstore.orderservice**

OrderProcess.java

- The Web Service interface class that is required for using the order process web service

OrderProcessServiceClient.java

- Class that provides a getServiceInterface method for the session controller to call when accessing the order process Web Services

3.4.6. In **CDStore/src** as part of **com.cdstore.orderservice**

navigation.properties

- A configuration file used by the session controller to determine the name of the next JSP file to load in response to a specific user action

3.4.7. In CDStore/WebContent/css

style.css

- Cascading Style Sheets file to define the visual aspects of various HTML elements

3.4.8. In CDStore/WebContent/jspf

footer.jspf

- Footer HTML content included on every page containing copyright information etc.

header.jspf

- Header HTML content included on every page containing the store name etc.

3.4.9. In CDStore/WebContent/view

cart.jsp

- HTML content that uses JSP to display the items currently in the user's shopping cart with options for removing items and changing quantities

cdstore.jsp

- HTML content that uses JSP to allow users to browse CDs, sort them by category, and add them to their shopping cart

checkout.jsp

- HTML content that uses JSP to allow users to view a summary of their order and confirm their order by entering their credit card information

resultpage.jsp

- HTML content that uses JSP to display if the order was processed successfully or not

3.4.10. In CDStore/WebContent/WEB-INF

web.xml

- Deployment description file for the CDStore web client that specifies the location of the index.jsp file, servlets, and constants such as the tax amount

3.4.11. In CDStore/WebContent

createaccount.jsp

- HTML content that uses JSP to display the account creation form

createaccountaction.jsp

- JSP page for processing the submission of the account creation form

index.jsp

- HTML content that uses JSP to allow the user to submit their username/password and log into the CDStore system

loginaction.jsp

- JSP page for processing the submission of the login information

4. Deployment

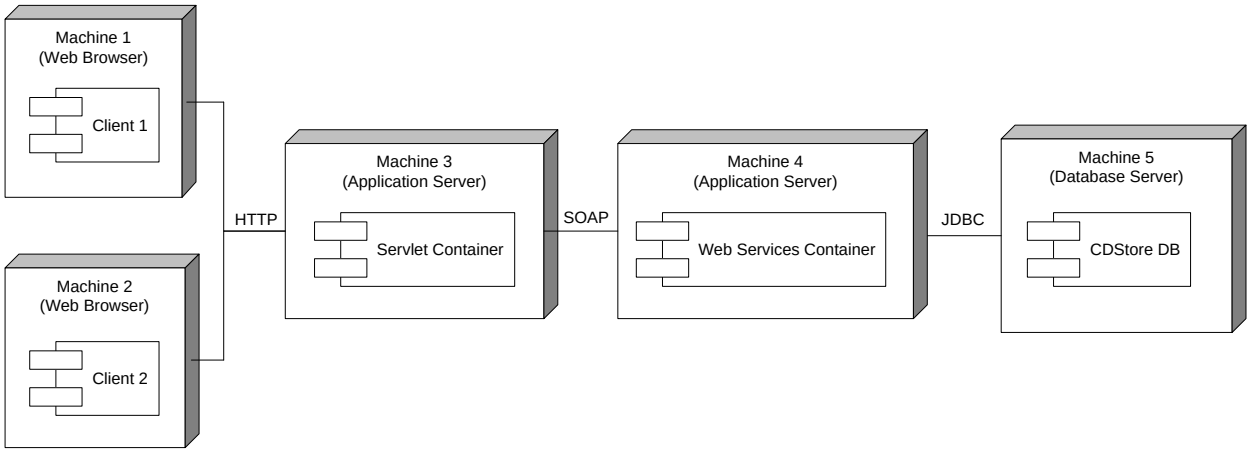

Figure 1: Deployment Diagram

As explained in the README.txt file, the project is deployed as four separate WAR files (CatalogService.war, OrderService.war, and CDStore.war) and one JAR file (Utilities.jar, which contains the dbagent, entities and shoppingcart packages). The user's web browser will connect to the application server running the CDStore.war, which requires Utilities.jar as a dependency. The Web Services consist of CatalogService and OrderService, which also require the Utilities.jar library. However, the Web Services can be deployed on a separate application server since communication takes place over HTTP (SOAP), as shown in Figure 1. In addition, the application server can be configured to connect to a separate machine running the Database Server. While the architecture supports such a wide deployment, it is possible to run the client web browser, servlet container, web services containers and CD Store database on the same machine. In test deployments, Firefox was used as the web browser, Tomcat as the application server, and MySQL as the database.

5. Packages and Architecture

Figure 2 shows the main packages that make up the DreamTeam CDStore web application. In the deployment, the PersistenceSubsystem and UtilitiesSubsystem are combined into the Utilities.jar file. The architecture, however, has the packages of the PersistenceSubsystem, WebServicesSubsystem and ClientDisplaySubsystem separated from the com.cdstore.entities and com.cdstore.shoppingcart packages. These two packages make up the UtilitiesSubsystem and are shared by the three other subsystems, showing its reusability. The ClientDisplaySubsystem features a Model-View-Controller architecture, whereby the SessionController (*the controller*) is responsible for receiving the user interactions from the JSP files (*the view*, shown in Figure 2 as the package com.cdstore.webview, although implemented in the WebContent folder) and obtaining the appropriate data from the DBAgent and Web Services (*the model*). The SessionController does this by using the CDStoreBean and OrderBean to communicate with the DBAgent via the twoWeb Services, namely ProductCatalog and OrderProcess. This is done through request SOAP messages that are generated based on the interfaces defined in ProductCatalogInterface and OrderProcessInterface, and response SOAP messages that are returned by the Web Services once they have contacted the DBAgent.

The SessionController is therefore responsible for actions such as updating a user's shopping cart with new items when then user has clicked the button to do this. The Filter Pattern is also implemented through the SessionFilter, which filters the access to different elements of the view so that users are not able to access the CD store page with the catalog without having first logged in.

As an example of reuse of existing code from other projects, the com.cdstore.shoppingcart package reuses most of the code from the example provided by Oracle Corporation [1].

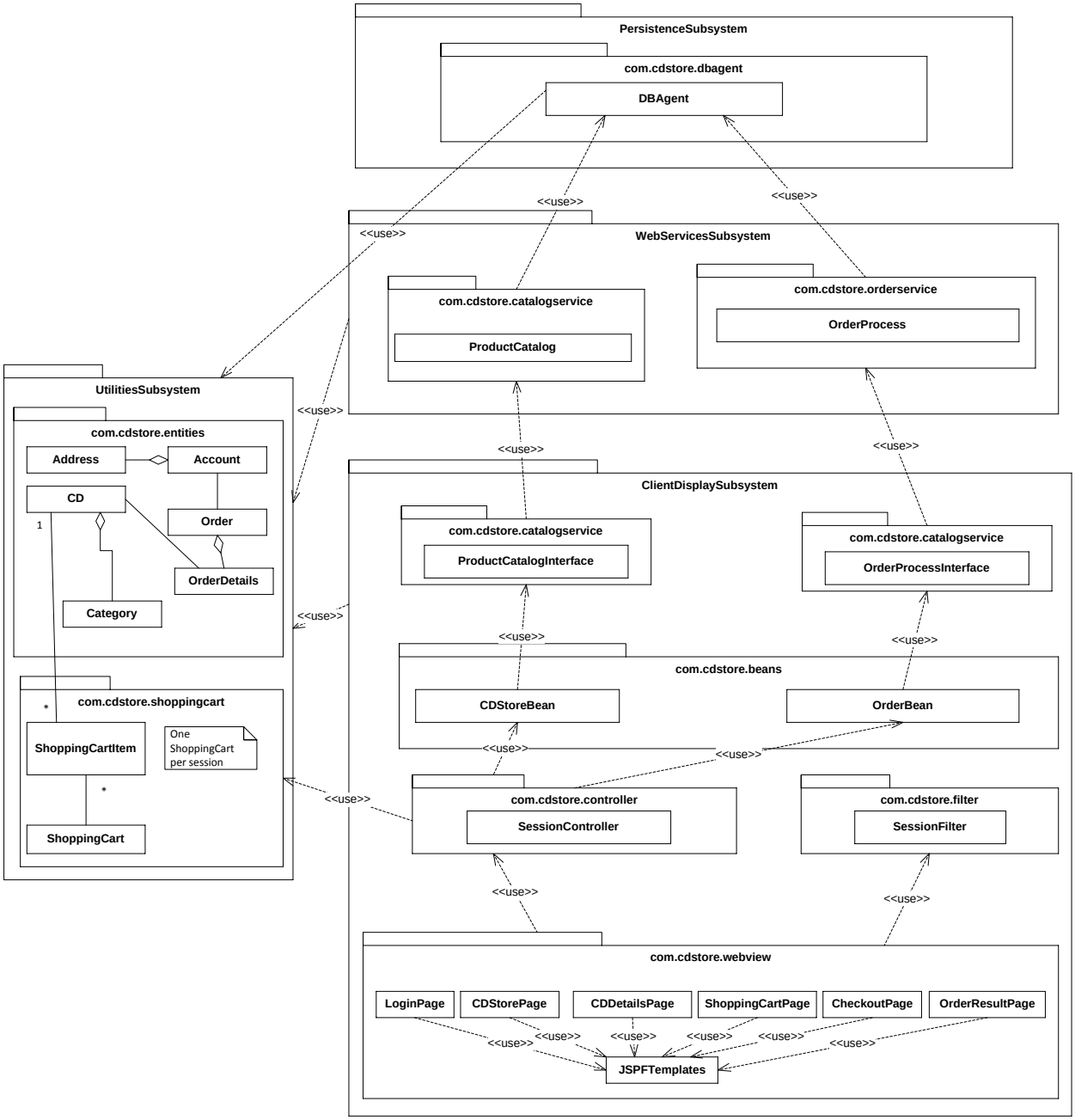

Figure 2: DreamTeam CDStore Packages and Architecture

6. Database Design

The com.cdstore.entities package previously shown in Figure 2 reflects the database schema shown in Figure 3. Of special note is how each “cd” is assigned a “category”, each “order” has an “orderdetails”, which has a “cd”, and each “order” is tied to an “account” which has an “address”.

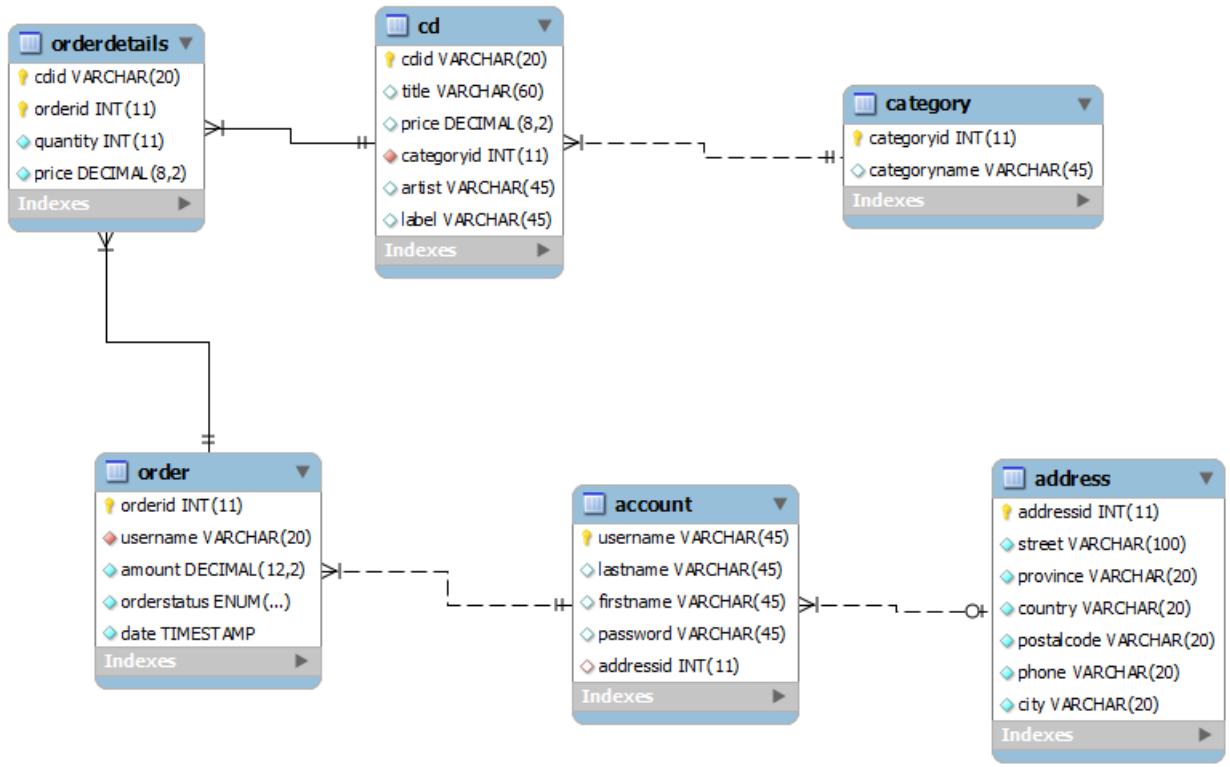

Figure 3: DreamTeam CDStore Database Schema

7. Sequence Diagrams

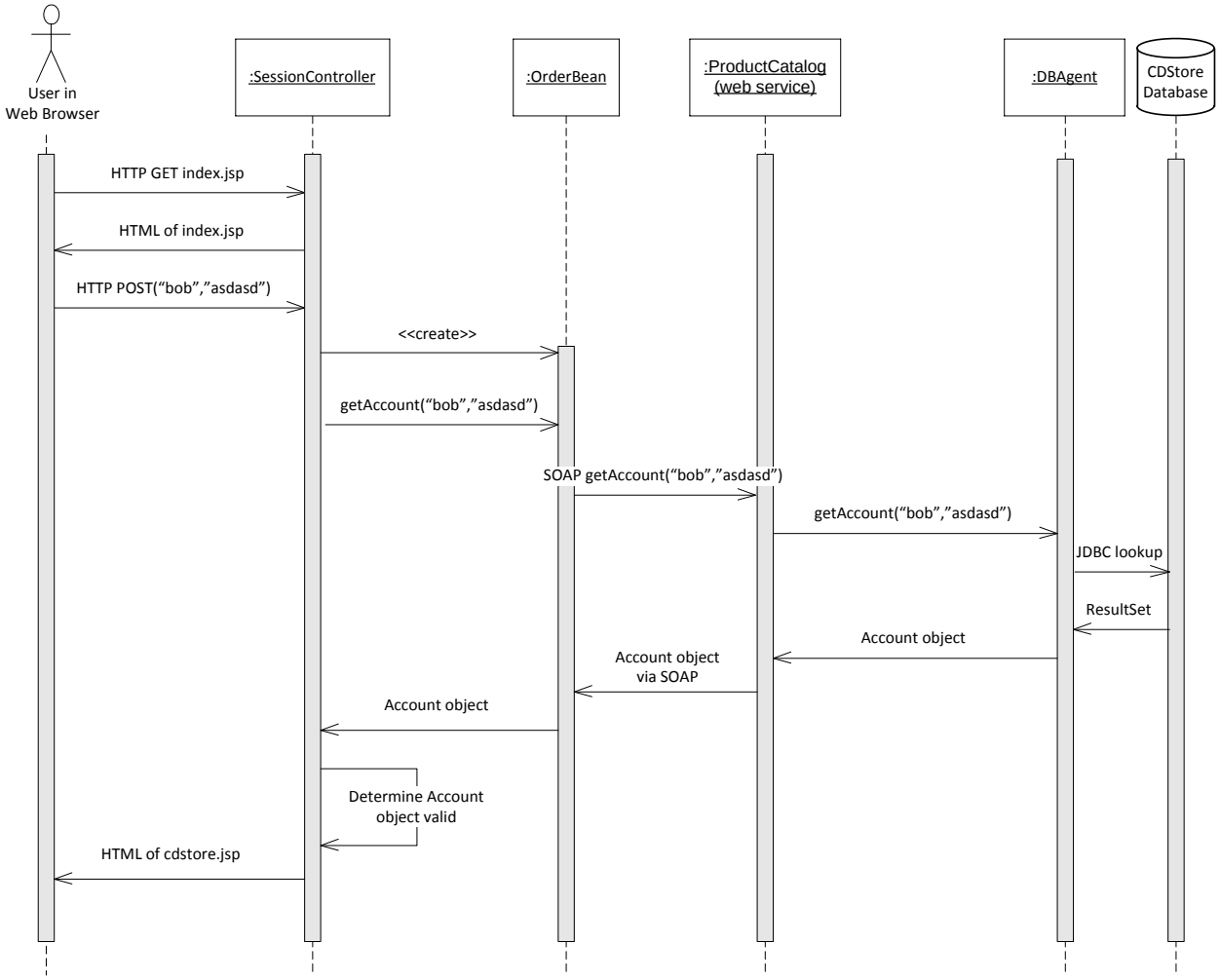

Figure 4: Interaction Diagram for Successful Login

The sequence chart of Figure 4 illustrates a typical login procedure. The user running a web browser performs a HTTP GET request for the index.jsp page, which is generated on the web server based on the servlets identified in the web.xml file; namely, the doGet method of the SessionController servlet. The user then enters their username/password and presses the "Login" button, which causes a HTTP POST message to be sent to the web server. This will be received and processed by the doPost method of the SessionController, which will create an OrderBean object for the session and use it to call getAccount on the ProductCatalog Web Service. The method call and parameters will be encoded in the SOAP format. Once received and decoded by the web service, the getAccount method in the DBAgent class will be called, which will query the relational database using JDBC. DBAgent will then receive a ResultSet object containing the query results, which will be transformed into an Account object (based on the entity) before being returned to the Web Service. The product catalog Web Service will then encode the Account object into SOAP and return it to OrderBean, which will pass it to the SessionController. The SesionController will determine if the object contains an error message or is a valid user. If the Account

object is valid, the HTML content of the cdstore.jsp page will be generated and transmitted to the user over HTTP.

Figure 5 shows an interaction with the ProductCatalog Web Service and a GET request triggering a database lookup. When a user clicks a category on the cdstore.jsp page, a HTTP GET request is sent to the web server containing the ID of the selected category. The SessionController then calls getCDList(categoryId) in the session's CDStoreBean instance. This request is delegated to the ProductCatalog Web Service via SOAP, and again to the DBAgent, which does a JDBC lookup in the database and returns a ResultSet. The ResultSet is then converted into an array of CD objects (based on the CD entity) and returned to the Web Service, which returns the array to the calling CDStoreBean. The CDStoreBean then returns the array of CDs to the SessionController, which inserts them into the HTML output and transmits them to the user's web browser.

Other scenarios, such as the checkout and confirmation process, involve writing data (rather than simply reading it) to the database by using JDBC, although the flow of events remains similar.

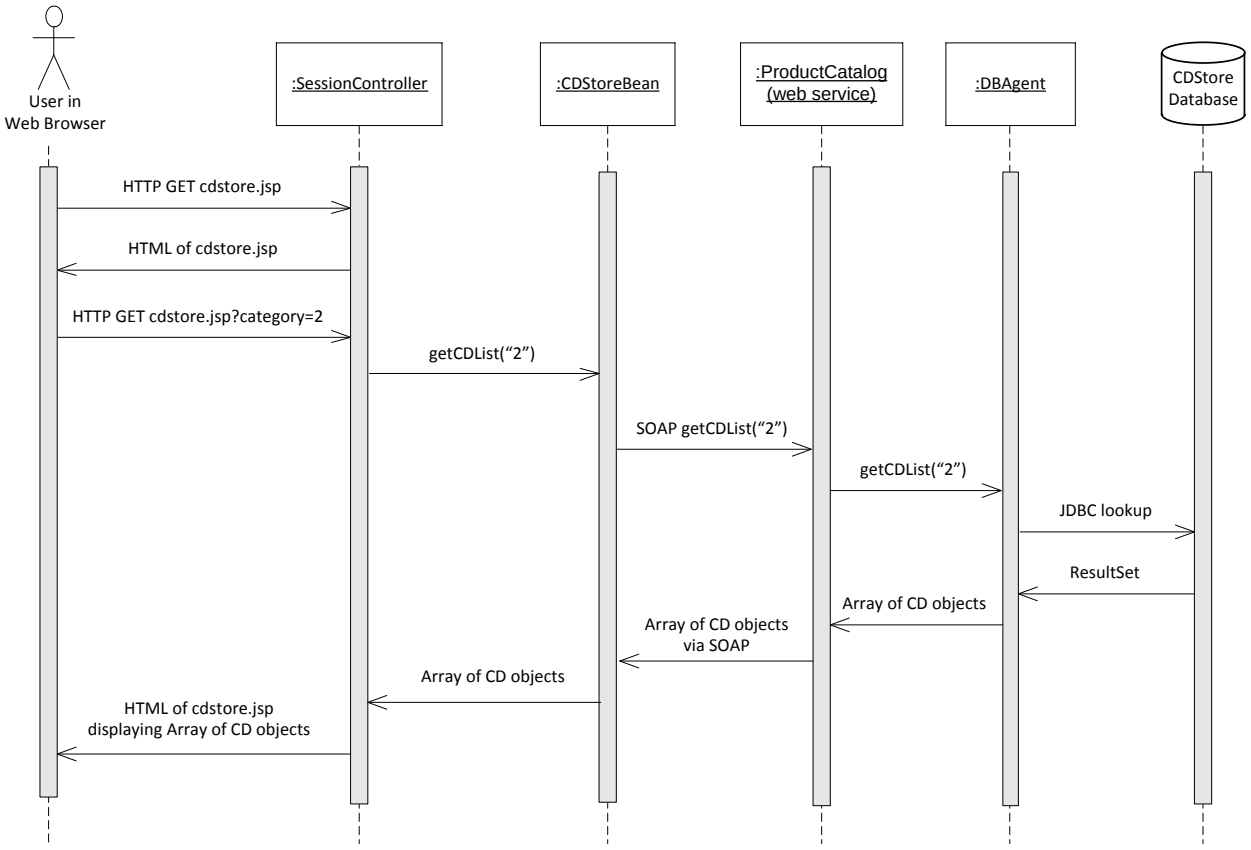

Figure 5: Interaction Diagram for Retrieving CDs in a Category

8. Screenshots

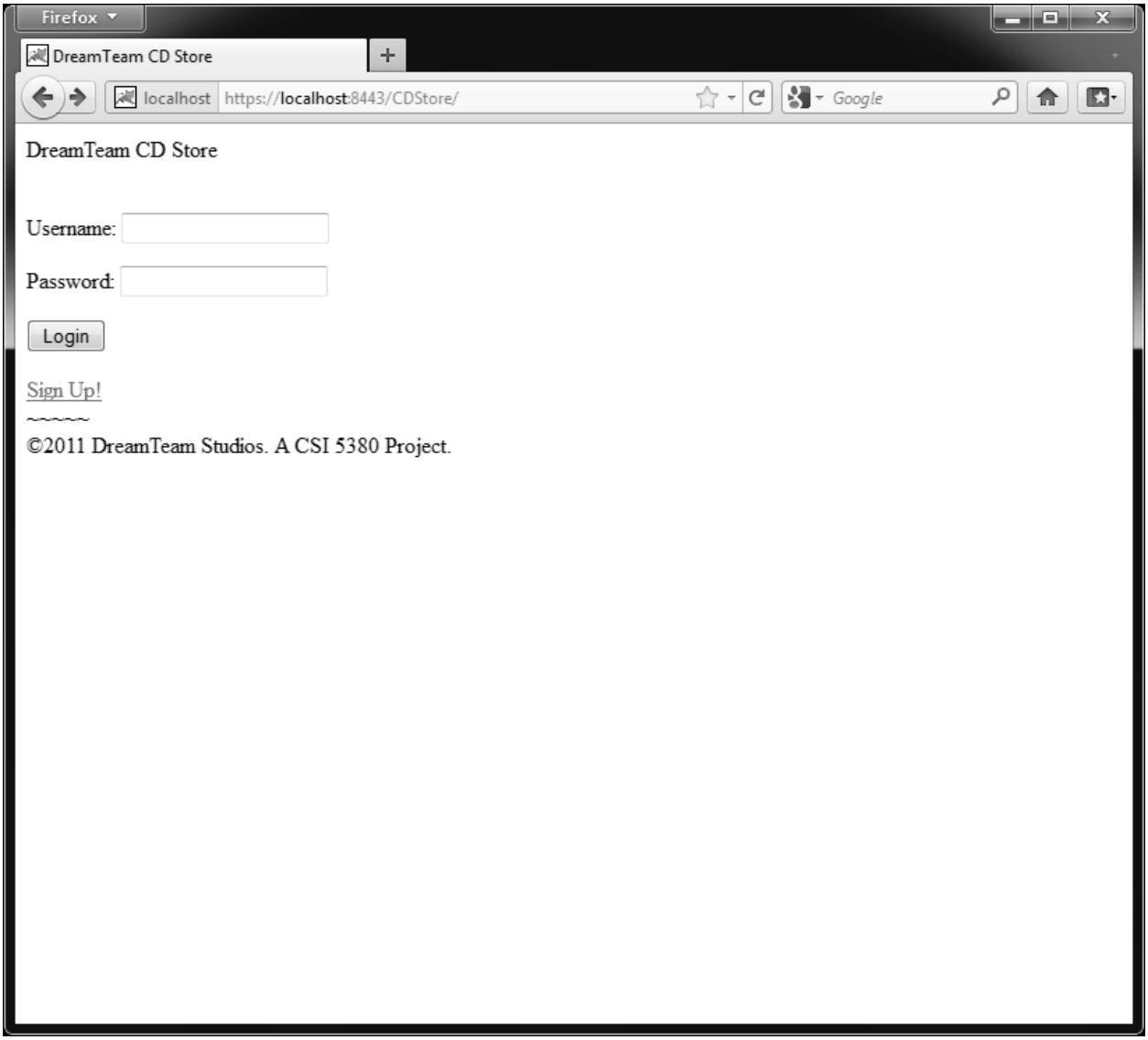

Figure 6: Screenshot of Login Page

Figure 6 shows a screenshot of the DreamTeam CD Store login page accessible by browsing to https://localhost:8443/CDStore/. As was illustrated in the message sequence chart of Figure 4, the page is based off of the HTML code in index.jsp and submits the username and password to the SessionController once the user clicks “Login”.

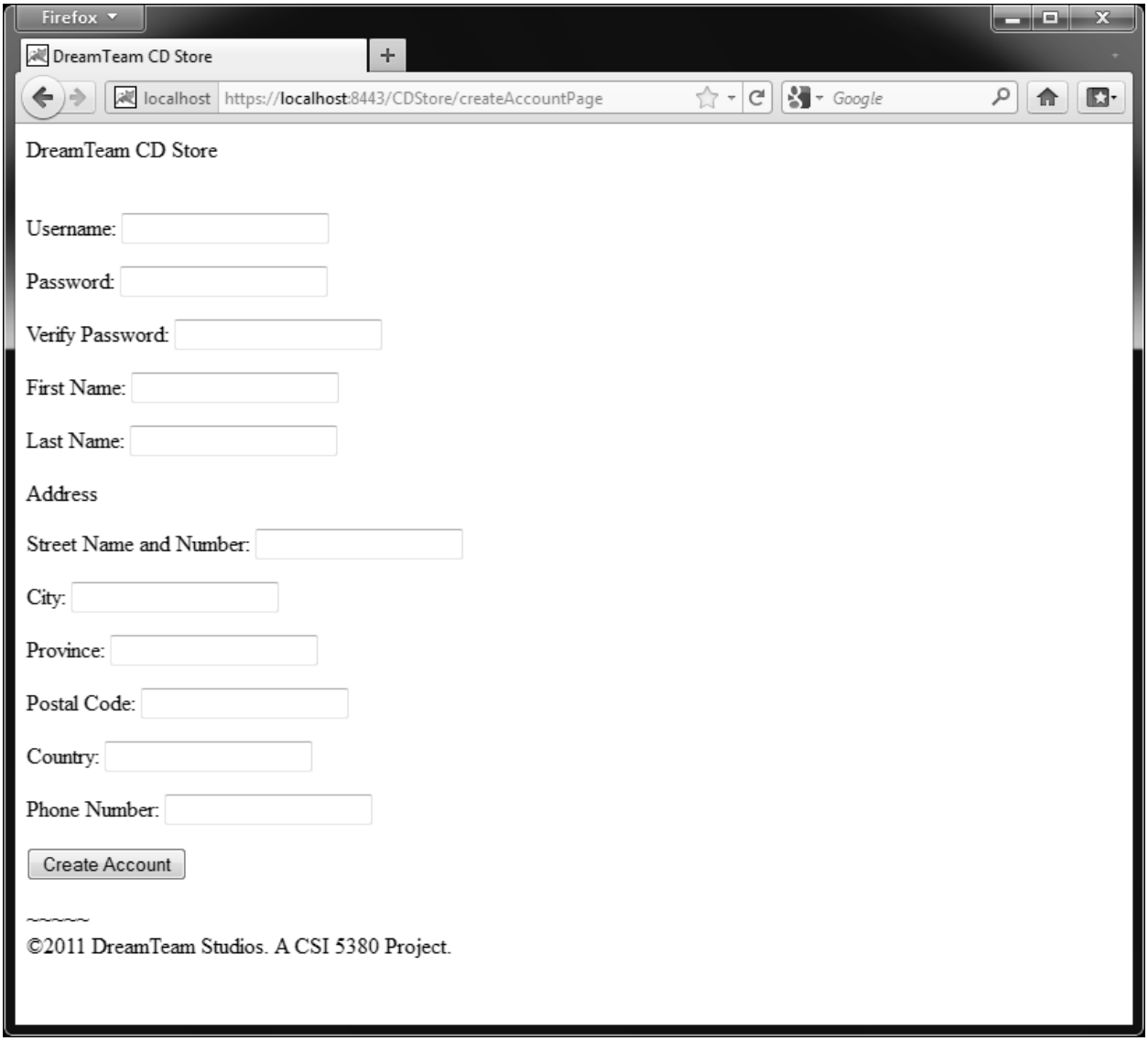

Figure 7: Screenshot of Account Creation

Figure 7 shows the account creation page (createaccount.jsp) that uses createaccoutaction.jsp to persist the user's account in the database and allow them to log in. Figure 8 shows the result when the user presses "Create Account" and the account was successfully created (username was not already taken).

Figure 8: Screenshot of the CD Store

Figure 8 shows the main CD Store interface that allows users to browse the CDs. The default view lists all CDs by querying getCDList() from the user's CDStoreBean instance to the ProductCatalog Web Service, which uses the DBAgent to fulfill the query. The SessionController then includes the resulting array of CD objects when sending the cdstore.jsp file (converted to just HTML) to the web browser.

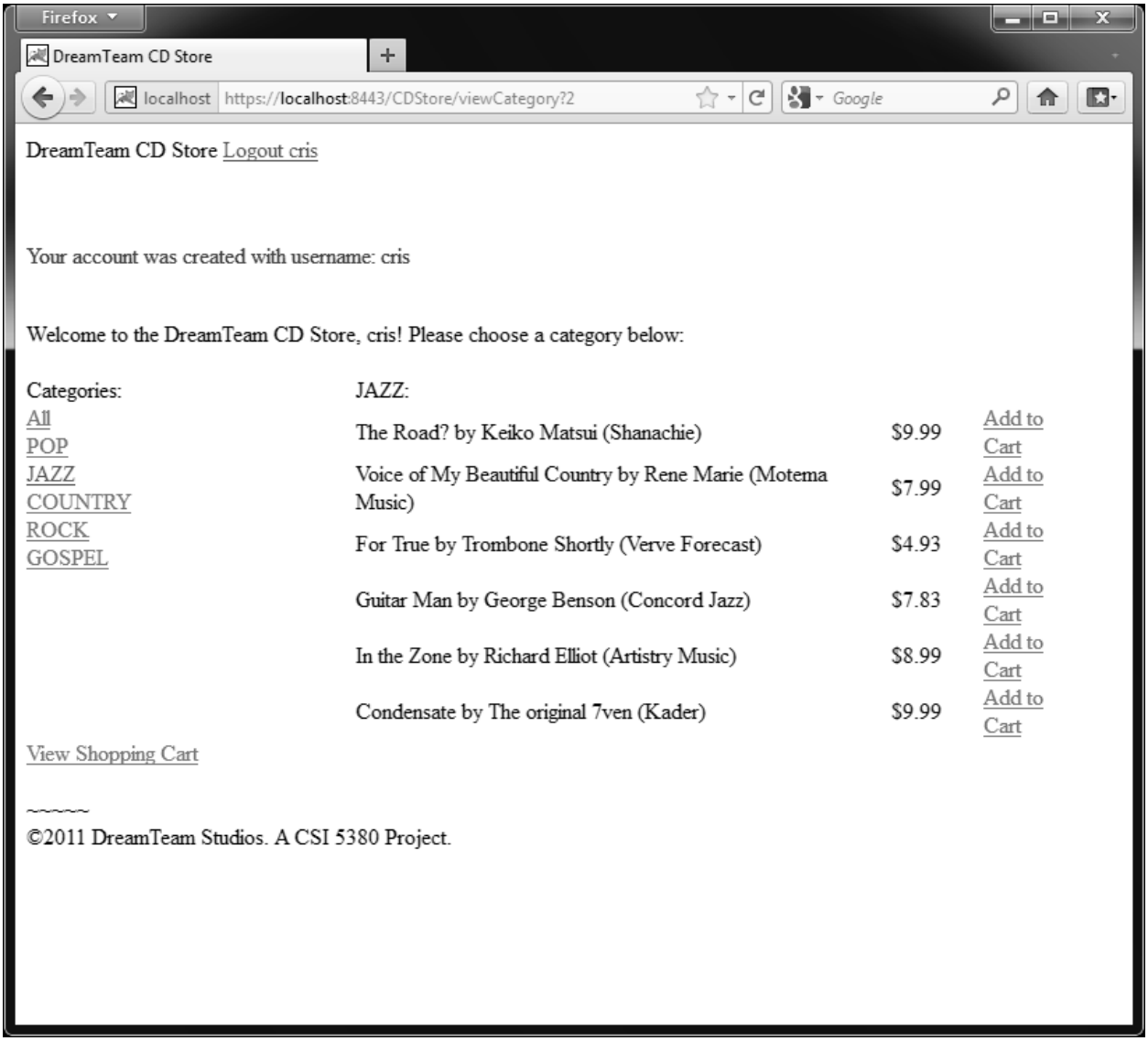

Figure 9: Screenshot of Category Selection

Figure 9 shows how the CD results have been filtered to just jazz CDs. The process that makes this possible was shown in the sequence chart of Figure 5. The user may click “Add to Cart” to select a CD they would like to order, or click “View Shopping Cart” to see the list of current items in their cart.

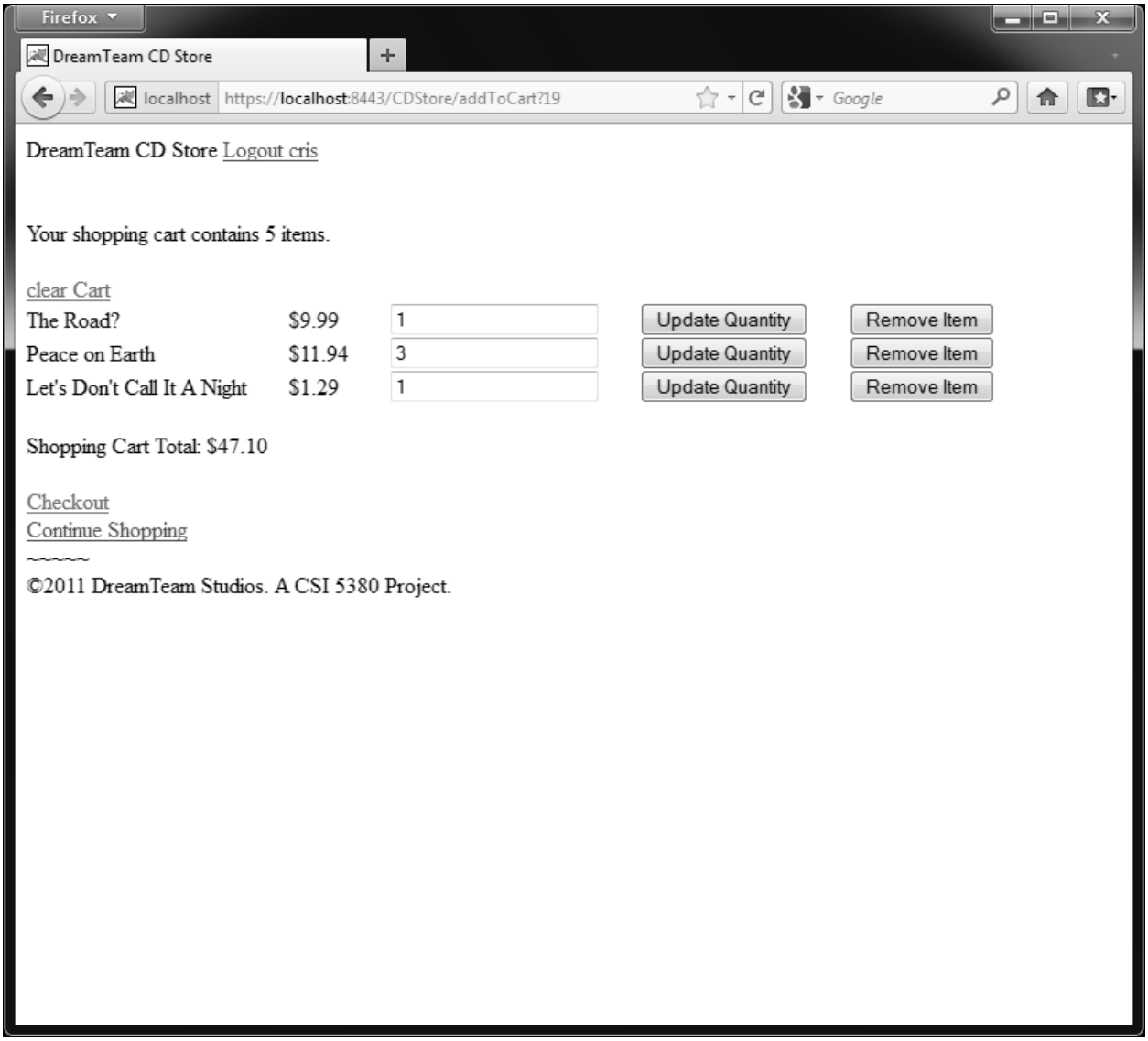

Figure 10: Screenshot of Shopping Cart

Once a user clicks “Add to Cart” or “View Shopping Cart”, they are brought to the shopping cart page shown in Figure 10. There is one instance per session of the ShoppingCart class (com.cdstore.shoppingcart in the Utilities project). A ShoppingCart, however, can contain multiple ShoppingCartItems, which contain the CD item information and quantity values shown in each row of the shopping cart summary above. The ControllerServlet computes the current total of the shopping cart for the user’s convenience. In addition, users are provided buttons to update the quantities of a particular item or remove an item entirely from the shopping cart. These actions cause the SessionController servlet to reload the cart.jsp page with the updated values and calculations. Since the shopping cart is not persisted, the Web Services are not required for displaying the shopping cart. Additional links are provided to clear the cart entirely or return to the CD Store page. Once the user is satisfied with their shopping cart, they can click “Checkout” to cause the SessionController to bring up the checkout.jsp page.

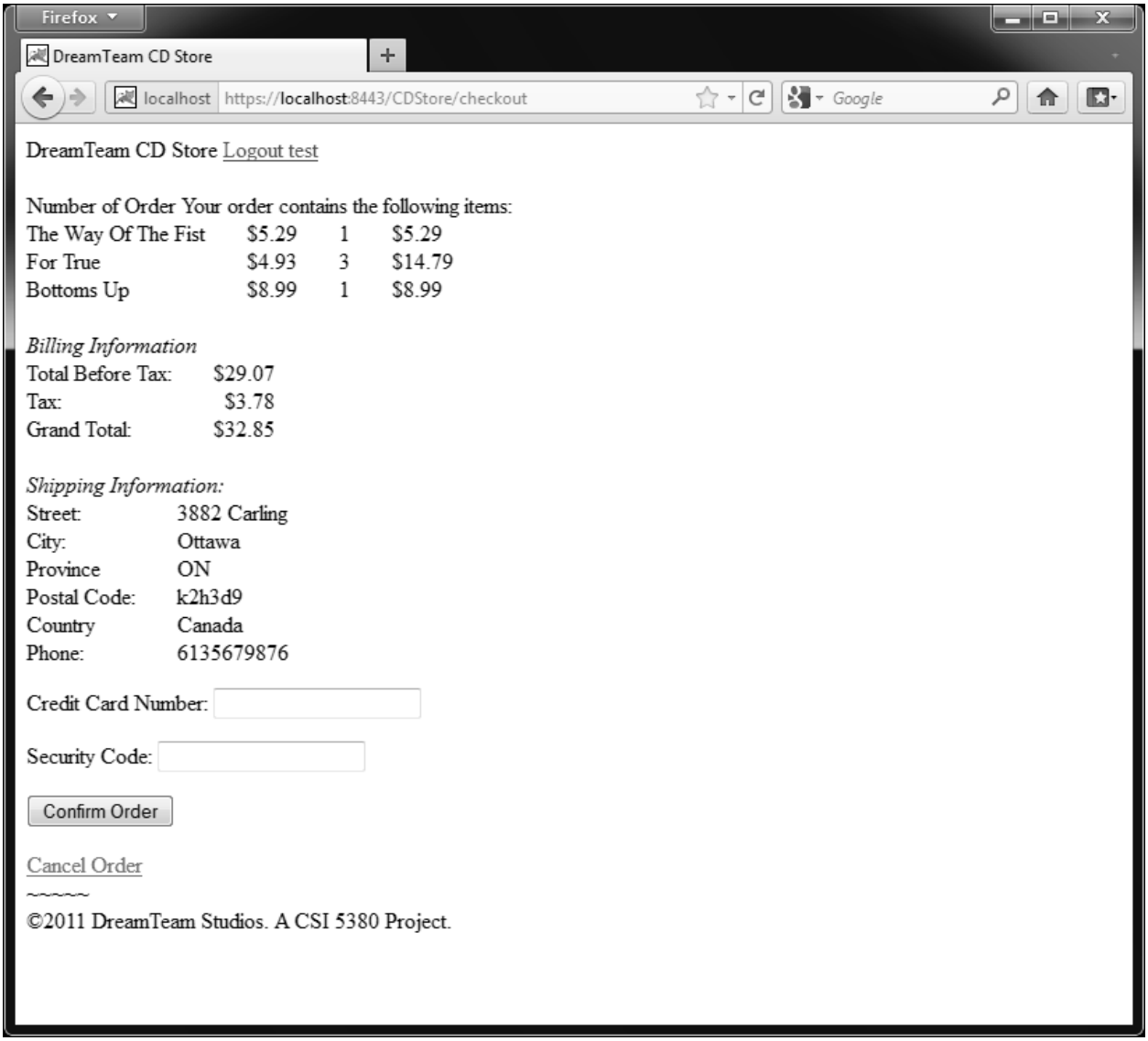

Figure 11: Screenshot of Checkout Page

The checkout page is shown in Figure 11 and summarizes the user's shopping cart, final amount they will be billed, address that their order will be shipped to, and finally provides the user with a place to enter their credit card information and credit card security code. Once the user clicks "Confirm Order", the order is persisted into the database through the DBAgent.

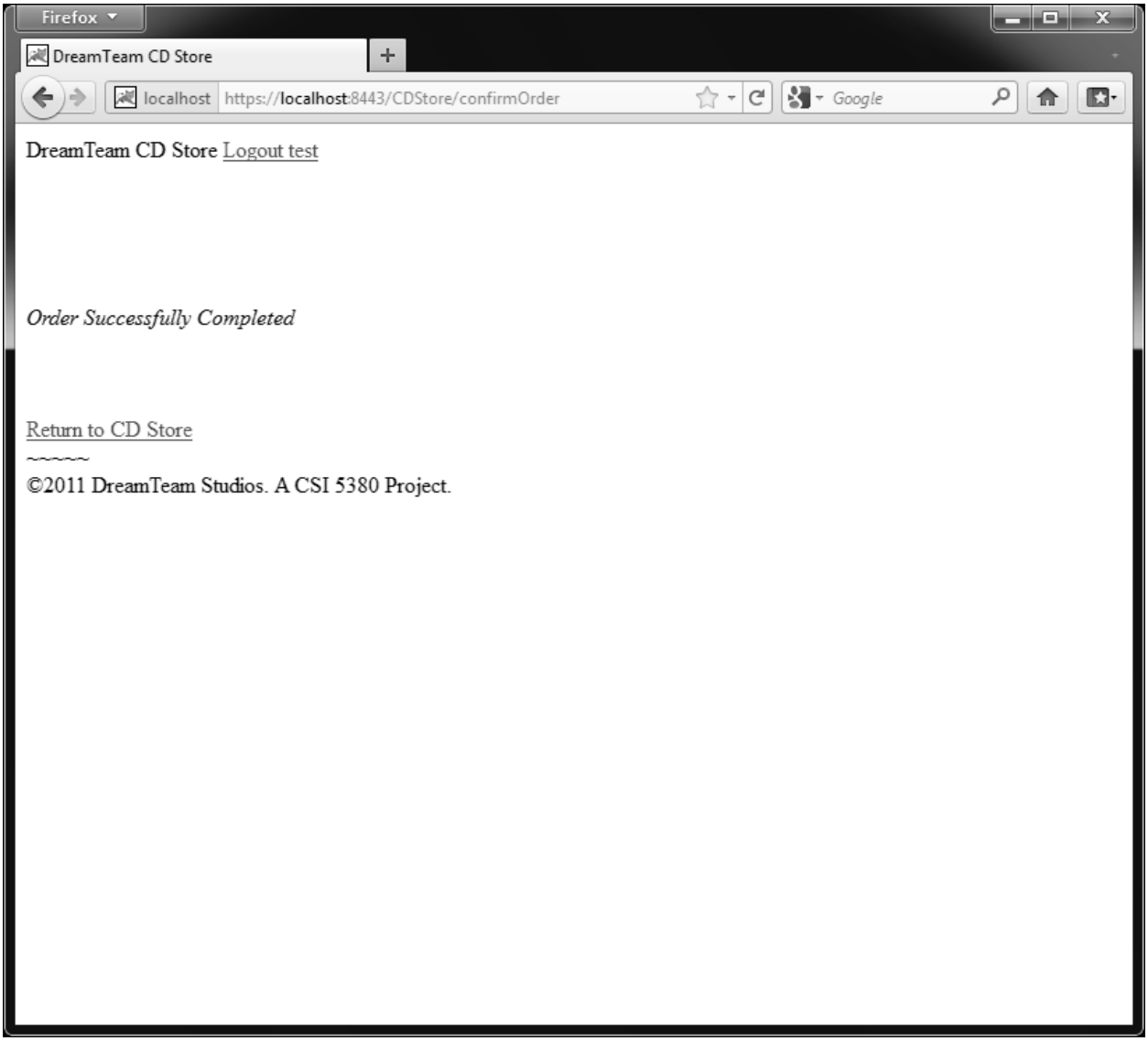

Figure 12: Screenshot of Result Page

Once the user has confirmed the order, they are presented with the result page which tells them if the order was successfully completed. Errors with credit card validation (such as the requirement to fail on every fifth order or problems creating the order via the OrderProcess Web Service) will also be displayed here. The user is given a link to return to the CD Store page or they may log out using the link in the header of the page.

9. Conclusion

By using technologies such as JSP and Web Services, an electronic commerce system was developed that allows customers to log in and browse a CD Store catalog, as well as submit orders for CDs. Future extensions could include JavaScript-based client data validation and a cleaner user interface. The main focus of the next part of the project, however, will be to create a Personalization Service with emphasis on product recommendation algorithms, cookie usage for obtaining various metrics about user browsing habits, as well as data mining and visual representation of the obtained values.

10. References

[1] Oracle Corporation. (2011). The NetBeans E-commerce Tutorial - Preparing the Page Views and Controller Servlet. Retrieved October 24, 2011, from http://netbeans.org/kb/docs/javaee/ecommerce/page-views-controller.html